Association Models Supplement to the Certified Wireless Universal Serial Bus Specification Frequently Asked Questions

Last updated: June 19, 2007

1. When using the numeric association model, how does the device know which is the correct host to associate with if multiple hosts are present and actively accepting new connections?

Manual user conditioning is required when using the numeric association model. Thus, it is expected that the scenario of a user wanting to associate a device while another host is also in the "accept new connections" mode would be extremely rare.

In the event that it does happen, the device would have to either choose a host randomly or simply fail out and force the user to try again at a later time. Because the hosts do not provide any identifying information beyond a host ID number, the device does not even have enough information to intelligently display a list of friendly host names and ask the user which is the correct one. We are considering adding additional pre-association information about hosts and devices to a future revision of Wireless USB.

2. What is the approximate CPU cost for Diffie-Hellman computations (in software)?

The computational cost of a modular exponentiation is proportional to the cube of the number of bits. Experience shows the cost of a 1024 bit modular exponentiation is about 4 million (2^22) instructions. Since

512 = 0.5 * 1024

1536 = 1.5 * 1024

2048 = 2 * 1024

3072 = 3 * 1024

etc.

you can estimate the approximate cost of a single modular exponentiation for these other sizes as

512 bit: (0.5)^3 * 2^22 = 2^19 = ~500,000 instructions

1024: (1)^3 * 2^22 = ~ 4 million

1536: (1.5)^3 * 2^22 = 27 * 2^19 = ~ 14 million

2048: 2^3 * 2^22 = 2^25 = ~ 32 million

3072: 3^3 * 2^22 = 27 * 2^22 = ~ 113 million

A Diffie-Hellman requires 2 modular exponentiation operations, so double these numbers to get an estimate for the "real" cost. Of course, if factory installed keys are used, then you only need one at run time, but then you also need at least the protection that a smart card provides for the private value used with the Diffie-Hellman.

As a side note, NIST recommends using at least 1024 bit operations from now up until 2010. Beginning 2010 they recommend using at least 1536 bit operations. Hence, our position is that algorithms being designed now will appear in products beginning in 2007 or so, and will be in the field 2010 and longer, so they should be designed to support at least 1536 bit operations.

RFC 3766, "Determining Strengths for Public Keys Used for Exchanging Symmetric Keys", is more cautious in their outlook and recommends a 1926-bit D-H key for 100-bit equivalent security and a 4575-

bit key for 150-bit security. By interpolation, the recommendation for 128b-bit equivalent resistance is a ~3000-bit D-H key.

The time to perform a single 3072-bit Diffie-Hellman calculation in software on a 200 MHz ARM9 processor is approximately 1/3 of a second.

Note that the Wireless USB Association Models specification has placed a limit of 2 seconds for the amount of time that a user can be forced to wait before doing any specific step of the association process. To meet this requirement, products with slow processors may need to start some of the calculations on power up and store this information until it is needed for a new association.

(Note: The FAQ needs additional performance information for other processors not already listed above. Please contact the FAQ authors if you can provide such information. Thank you!)

3. What is the approximate gate count for hardware implementations of the cryptographic functions required by Wireless USB association?

The Montgomery exponent function (used to compute the Diffie-Hellman exponents) is between 20k and 50k gates.

A SHA-256 implementation is approximately 22k gates. To process a 512-byte digest requires approximately 65 clocks. To add HMAC-SHA-256 requires an additional 5k gates above and beyond the SHA-256 implementation (total 27k gates for both SHA-256 and HMAC-SHA-256).

Total gate count to support SHA-256, HMAC-SHA-256, and Diffie-Hellman is ~52k gates.

Note that these gate requirements do not apply to the cable model.

4. Why isn't NFC being used for association?

NFC, or Near Field Communication, is a short-range (10 cm or less) wireless technology that operates in the 13.56 MHz band, a frequency approved for world-wide license-free use. NFC uses magnetic induction over a contactless interface to achieve bidirectional communication.

NFC is a very promising technology with tremendous potential for simplifying the initial configuration of wireless devices. It is being actively investigated for use as a future Wireless USB association model. NFC was ranked first out of numerous association model methods in a May 2005 Wireless USB focus group study.

5. Was IR (infrared) considered for initial association?

Infrared was initially considered but was eventually abandoned. We investigated two forms of IR, "CIR" (Consumer IR, used in CE devices) and IrDA (used predominantly in the computing industry). CIR is low-cost but did not have sufficient speed for our data transfer requirements. IrDA was fast, but the cost and complexity was higher. In both CIR and IrDA, the cost and manufacturing implications of having an optical transceiver module were prohibitive. The possibility of an eavesdropper being able to detect stray infrared emissions was also raised as a security concern with all IR models.

6. The numeric model requires a button, an LCD, and an LCD driver. These parts add cost and complexity to devices.

The numeric model does have additional hardware requirements. In designing the security for Wireless USB, it was determined that an LCD was necessary to provide an adequate level of protection against attacks when using the numeric model. Please note that the numeric model is not required for devices. Manufacturers who are concerned with the implementation cost of the numeric model should investigate one of the other models available.

7. There are too many Wireless USB association models. There should be only one model that everybody has to implement.

In an ideal world, we would have one model.

Many Wireless USB devices will also have wired USB functionality, which leads us to implement a cable model. This allows a device to be automatically and seamlessly configured wirelessly if the user already has it plugged it in.

Since requiring a cable is not acceptable to everybody, we also need to support at least one additional non-cable model. From our focus group testing, we think that NFC is the best solution of the consumer-tested approaches, but we think it is not quite ready yet and need time to research it as a solution.

Thus, we need another solution and have chosen the numeric model to be that solution.

We are not concerned about customers being confused by multiple association models: To protect the value of the Wireless USB brand, hosts will be required to support cable and numeric, while devices are free to choose either. Because the USB topology allows only one host talking to multiple devices, this means that a device will always be able to connect to a host regardless of which model the device chooses to implement. Note that embedded hosts are treated differently and only have to support the association models used by the devices on the embedded host's targeted peripheral list.

We think that our current proposal is a reasonable compromise given the practical constraints.

8. Why can't we do a "button-push" model? Some products on the market today just use button pushes and it seems to work.

While compelling from a usability perspective, relying only on button pushes without a separate out-of-band confirmation step is very weak from a security standpoint.

Systems that use button pushes typically rely on one of the following methods for providing association:

Completely insecure models simply use the button pushes as a way to put the products into a mode where they can find each other. Attackers can simply insert themselves at will since there is no security. Even in benign situations, users have to be careful not to try to introduce too many products at once as they will interfere with each other.

Fixed-key models rely on pre-agreed-upon keys that are programmed into the products at manufacture time. When the buttons are pushed by the user, the products will temporarily use the pre-configured key to transfer the secret information. In simple implementations (often used to reduce manufacturing costs), the pre-configured keys are globally shared across all products. The disadvantage of this model is that if the globally shared key is compromised, then security for all products that use that globally shared key is similarly compromised.

In timing models, the user is instructed to push buttons on two separate products within a certain period of time. This model provides quite low security as the human perceived response time is several orders of magnitude slower than the speed at which automated computer attack systems can operate.

The unfortunate reality is that all cryptographic systems need a secure out-of-band channel of some sort (for example, a user confirmation) for their initial configuration. Button-push methods without some form of out-of-band verification or user confirmation do not meet the minimum security requirements for Wireless USB.

9. How is disassociation defined, such as in the case where an attacker plugs a device into the host, or if a previously associated device is lost?

Disassociation is at the manufacturer's discretion. From a technical perspective, disassociation is quite easy: The host or device simply has to erase its stored connection context, and subsequent communications with the paired host/device will be impossible until a new association is completed.

In the case of a lost device that had been previously associated with a host, the owner should be given some way to list active associations on the host and then offered the ability to remove one or more as necessary. Because the exact implementation of this procedure will vary from product to product, the decision on how to achieve the best user experience and security for their customers is left to the manufacturer.

The case of an attacker plugging a device into a host: One of our design goals in Wireless USB was to provide security equivalent to USB. This means that we do not defend against any attack that would

succeed against USB today. Once physical access to a host or device has been compromised, we make no guarantees about security in either wired or Wireless USB. Having said this, we highly recommend (and in some cases require) active user confirmation when adding new devices to a host. Most software manufacturers are also implementing requirements above and beyond the specification requirements to help avoid such security vulnerabilities when hosts are locked. When combined with host security features (such as locking the screen of a PC when unattended), this should stop most Trojan horse style attacks.

10. What happens if a device discovers a valid host in range, but its connection context for that host doesn't work any more?

What to do in this scenario is left up to the product manufacturer. However, generally, the device should indicate to the user that it is unassociated. The user would have to re-start the association process.

11. Are temporary associations that time out after a specified period supported?

The Wireless USB protocol supports "one-time" associations, but otherwise does not make any special provisions for temporary associations. A host or device vendor is free to add such functionality into their products. From a technical standpoint, a host can disassociate a device at any time by simply removing its connection context from its list of valid associations. There are numerous usability considerations that need to be addressed, of course.

12. Are vendor-specific association models allowed (in addition to the mandatory models)?

At this time, a final decision has not been made. However, vendor-specific association models are strongly discouraged and may be forbidden by the logo certification process in the future.

13. What does a software implementation of SHA-256 look like?

An ARM implementation of SHA-256 is (code size) 256 bytes of table + 260 bytes of ARM assembly for the core SHA-256 transformation. Blocking, padding, and byte-swapping overhead are extra on top of that.

14. What is the size of the random numbers A and B used in the Diffie-Hellman protocol?

The secret integers A and B must be greater than or equal to 2 and must be at least 256 bits long. They should be chosen randomly from the entire available number space, which is [2, 2^256-1].

15. What are the rules and regulations regarding the export of products containing strong cryptography (like AES) from the United States?

The Bureau of Industry and Security web site from the U.S. State Department (www.bxa.doc.gov) is a good resource for information about this subject, as is the Wassenaar Arrangement on Export Controls for Conventional Arms and Dual-Use Goods and Technologies (http://www.wassenaar.org/index.html).

The Thomsen and Burke law firm (www.t-b.com) in Baltimore specializes in cryptography and may be a good place to start. Their web site has an entire section devoted to cryptography (http://www.t-b.com/15.html), including checklists, a license matrix, and an FAQ.

Generally, a vendor must register with the government by filling out and submitting a form describing any product that uses strong cryptography. There is no approval process and registration does not imply that the government will approve or deny the product. Please consult the BXA web site or a law firm like Thomsen and Burke for more detailed information.

16. What are the rules and regulations regarding the import, export, or use of products containing strong cryptography in countries other than the United States?

The Wassenaar Arrangement on Export Controls for Conventional Arms and Dual-Use Goods and Technologies (http://www.wassenaar.org/index.html) is a good source of information for international rules and regulations for many countries. Note that China, Singapore, and Israel do not participate in the Wassenaar Arrangement.

17. We applied for export for AES and Diffie-Hellman, and the NSA came back with restrictions on sales of source code to foreign governments (like China and India) and companies they fund.

Products that contain AES and/or Diffie-Hellman and are sold outside the country are subject to additional government restrictions. We have been advised that these rules do not apply to source code that is made freely available. However, companies are encouraged to consult a law firm with expertise in this area, such as Thomsen & Burke (www.t-b.com).

18. How long does the association process take?

The Association Model Specification requires that a user never has to wait more than two seconds for any one step of the association. The entire association process may take longer than two seconds.

19. What is an in-band association model?

In-band association uses Wireless USB protocol to transmit the Connection Context from the host to the device. When using in-band association, steps must be taken to ensure that keys are privately exchanged and that there are no man-in-the-middle attacks since all transmissions are over an uncontrolled medium. One example of in-band association is the numeric association model. However, the numeric association model is not an "in-band" model in the purest interpretation of the term because it requires an out-of-band verification channel (i.e., the manual user verification of the displayed numbers) to ensure security.

20. What is an out-of-band association model?

This type of association does not use the Wireless USB protocol to transmit the Connection Context information from the host to the device. Often, but not always, transmissions made out-of-band from the Wireless USB protocol are secure in their own right and do not require special protection from eavesdropping. One example of this is the cable association model.

21. Was MQV considered for Wireless USB association?

MQV is a cryptographic method based on Diffie-Hellman that uses elliptic curve technology. For more information, see http://en.wikipedia.org/wiki/MQV.

MQV assumes that public signature keys have been distributed via an out-of-band channel and verified as belonging to the correct entity. Thus, MQV does nothing toward removing the need for an out-of-band channel (in fact, no known cryptographic algorithm does).

Second, MQV requires a significant performance step if it is to provide any security. Let G be group of order p with a subgroup of prime order q generated by g, with gcd(q, p/q) = 1. The MQV public signature key for A is g^a for some secret a, and the MQV public signature key for B is g^b for some secret b. The significant computational step needed for the security of MQV is B must verify that g^a is of order q in G, and A must verify that g^b is or order q in G. That is, B must verify that g^a, (g^a)^2, (g^a)^3, …, (g^a)^(q–1), (g^a)^q = 1 are all distinct (there are actually more efficient tests, but they are still computationally onerous), and likewise for A. This seems like an unwarranted cost for what is used as a one-time setup key.

Third, MQV is covered by patents and would most likely require royalty payments for use.

For these reasons, MQV is not seen as providing significant value over the current solution for Wireless USB Association.

22. The association supplement specifies that random numbers must be derived from physical sources of entropy. The Wireless USB and WiMedia documents talk about generating random numbers using AES-CCM. Is there a discrepancy?

Section 6.5 of the Wireless USB Specification 1.0 provides guidance and sample code for generating random numbers using the AES-CCM function. The techniques described in section 6.5 for generating random numbers are completely valid for use in Wireless USB association. Please note that the term "randomness sample" which is used throughout section 6.5 must be interpreted to be derived from physical sources of entropy.

Note that using the AES-CCM function is not required for generating random numbers that are used in association. If the methods described in section 6.5 of the Wireless USB Specification are not available for some reason, designers are free to use their own methods as long as they comply with the random number requirements as specified in section 3.6 of the Association Model Specification.

23. At some point in time, a "serial number" or "PIN" model was under consideration for Wireless USB association. This method is also used by some other wireless technologies. Why was this model removed?

Much consideration was given to the possibility of using a PIN-type number that would be printed or displayed on the device and simply entered by the user on the host. Although the process of reading a number from a device and entering it onto a computer is well understood by consumers, it has substantial limitations for Wireless USB:

First, this would require that all Wireless USB hosts would need to provide a way of entering the number. This is a significant burden for limited hosts and dual role devices.

Second, printing a unique number on a sticker and then synchronizing the device logic to match that unique number was seen by some as a manufacturing burden. Devices would no longer be able to be manufactured generically.

Third, the security model of a fixed number printed on a sticker is weaker than the numeric model that was ultimately adopted (the security of the fixed number was further weakened by our usability desire that the number be 4 digits long). Several attacks against the PIN model were identified that would compromise the security of a device given a sufficiently motivated attacker. These attacks do not exist in the numeric model.

Fourth, in usability studies, a slight preference was shown for the numeric model (in which two short numbers are compared) versus the PIN model (in which a longer number is manually entered by the user).

Finally, there was concern about any stickers on the device becoming worn out such that the numbers could no longer be read, or that a device manufacturer would include the number in the user manual or warranty registration card, which would quickly become separated from the device and cause customer frustration.

24. Why does the AM spec require random numbers to be greater than or equal to two?

The cryptographic algorithms used by the Association Models specification do not work or have security vulnerabilities if a zero or a one is used for a random number. Thus, random numbers are required to be non-negative and greater than or equal to two.

25. What is the purpose of the hash commitment in the first stage of the Numeric Model?

Hash commitment is a process where a device sends a hash of its public key to the host before it actually sends the public key. The hash commits the device to the device's public key value, without revealing the device's public key until later, after the host's public key is revealed.

With hash commitment in place, a man-in-the-middle attacker would have to find a public key that creates a hash value that matches the hash commitment. This task is not computationally feasible due to the nature of the hash function used (SHA-256).

Use of a hash commitment thwarts man-in-the-middle attacks with a much smaller verification string, thereby improving the usability of the process.

When using hash commitment, the user must be asked to verify 2 to 4 numeric (decimal) digits. This means that an attacker's chances of success would be between 1 in 100 and 1 in 10,000.

26. In previous discussions about association, the concept of using audio or blinking LEDs was mentioned, but these are not mentioned in the current specification.

While this solutions were briefly discussed as possible ways to meet the needs of Wireless USB association, ultimately they were not selected for various reasons.

27. The Numeric Model has four message exchanges (M1-M4) and the Wireless USB spec refers to a four-way handshake. Are these referring to the same thing?

No. The message exchanges in the Numeric Model represent a Diffie-Hellman exchange that is used for association, whereas the four-way handshake represents the derivation of a pair-wise temporal key (PTK) that is used for encryption of payload data.

28. When using the cable model, the host will load the appropriate driver for the association interface. Does the host also enumerate and load the driver for the device itself?

Yes. The host will enumerate and start using the underlying device as soon as it is connected with the USB cable. This is done independently and in parallel with the association.

29. If a device only supports the Cable Model, what should the device do if it is powered on before it is connected to a host with a USB cable?

Unless the device has support for the Numeric Model, it should not attempt to connect with any hosts over the Wireless USB channel. Although technically it is possible the device to start the "new connect" process, to do so would be pointless since the device would be unable to finish the association process. The device should wait until it has been associated to a host using the Cable Model and only then begin normal Wireless USB operation.

30. In the Cable Model, if the host finds that it already has a connection context for a device trying to associate, the Association Models document says that it "sends a new CC to the device with a freshly-generated CK." Why isn't the existing CC used.

The Cable Model was designed this way to address the rare but possible case where a device's CK may be out of sync with the host's copy. For security reasons, the only information exchanged between the host and device during the first phase of the Cable Model is the CHID and CDID. Thus, the host has no way to know whether the device's CK is valid or not. To ensure that the device's CC fully matches the host's, the host always sends a new CK to the device.

Also, a basic cryptographic principle is to rotate keys as often as possible.

31. I have seen a security solution in which the user can push a button on each side between 1 and 32 times, and the number of button presses serves as an authentication for the Diffie-Hellman exchange. How does this compare to the Wireless USB Numeric Model?

First, it is expected that the usability of this multiple-button-press approach would be very poor. This conclusion is based on a May 2005 Wireless USB focus group study in which numerous association methods were tested. In that study, a similar process (involving blinking an LED a certain number of times on each side) was shown to the participants. It received the lowest scores of any of the methods tested, and also received many negative comments. Hands-down, users preferred simply verifying or entering a PIN number or using NFC technology.

Second, the number of bits of entropy that can be generated by having a user press a button a certain number times is very low. For between 1 and 32 button presses, the number of bits of entropy is $\ln(32)/\ln(2) = 5$. This level of security is almost the same as a simple unauthenticated Diffie-Hellman exchange. Moreover, it is unlikely that users will choose to press the button more than 16 times (or even 8 times), further reducing the number of bits of entropy to 4 (or fewer).

32. Why are hosts required to implement both the Cable and Numeric models?

The motivation in requiring hosts to implement both the Cable and Numeric models was to ensure that customers will never be in a situation where they can't associate a host and a device. For example, consider a camera (host) that only supported the Numeric Model. A hard drive (device) that only supported the Cable Model would be unable to connect. This would create a poor perception of Wireless USB as a universal connectivity solution.

The solution to this problem was to require hosts to do both the Cable and Numeric models, thereby ensuring that all devices will be able to connect to all hosts.

A notable exception is embedded hosts, which are only required to support the association models that are used by the devices on the embedded host's targeted peripheral list.

33. The association models supplement refers to a "Cable-Based Association Framework" document, but this document does not appear to exist anywhere.

The Cable-Based Association Framework was originally intended to be published as a separate specification. For various reasons, this has been delayed and will probably not happen in the near future.

Please be assured that this has no effect on the Association Models supplement or Certified Wireless USB. Everything necessary for the correct operation of the cable association model is included in the Association Models supplement that has been published.

34. Was the possibility considered of using the Certified Wireless USB radios in a low power mode for association? The idea being that in low power mode, attacks would be very difficult and there would be no additional cost because the existing radio is being repurposed.

Much consideration was given to trying to do an "in band" association using the Certified Wireless USB by putting the radio into a low power mode. Ultimately, it was decided that the risk of attack was too great with this solution.

There are two attacks to consider here, eavesdropping and active attack. In the case of eavesdropping, it is easy to see that an attacker with a large receiving antenna would have no trouble in detecting very faint RF emissions even from great distances. In the second scenario of active attack, an attacker with even a

modest budget can readily piece together off-the-shelf components to produce an attacking device that is more than capable of confusing a low-cost, consumer-grade product into thinking that it was much closer than it really is.

35. What is the purpose of the bmBandGroups value sent in the cable and numeric models?

The Wireless USB 1.0 specification requires products to use band group 1 only. However, it is expected that at some point in the future, products will support additional band groups. As a result, both the cable and numeric models have been designed to be flexible enough to make association easier for hosts and devices that support band groups other than band group 1.

36. In the numeric model, what happens if the number displayed on the host's screen does not match the number displayed on the device's screen?

When the numbers displayed don't match in the numeric model, this indicates that the host is not connected to the device that the user thinks it is. In this case, the user must press a "don't match" button on both the host and device. If a "don't match" button is not available on the product, then the user should wait for the product to time out or power cycle the product.

37. Do we need some restrictions how to choose a random seed in order to avoid using the same public key previously installed in the other device or accidentally derive the same public keys among the manufacturers? (e.g. seed = manufacturer ID + date + time)

If the guidelines in section 3.7 are followed, no further special steps are required in order to guarantee that public keys are unique. Note that public keys are specifically prohibited from being installed at manufacture time by the specification and must be freshly generated at run-time by the device.

38. Is it mandatory that the private key is stored in a secure memory which is implemented in the Wireless USB chip inside? Or can we use non-volatile (i.e. two-wire EEPROM) secure memory which is implemented outside of the chip to store the private key?

The private key information can be stored wherever the manufacturer determines is the best location to protect the information.

39. Suppose that a device finds more than one host that is accepting new connects – in this case, there is very little information available to the device and the device has to choose the host based only on host ID. (This question applies to the numeric association model.)

Just because a host is within range of a device does not mean that the host is available for new connects. It is expected that by requiring hosts and devices to be conditioned to accept new associations, that the situation where a device has to choose between two or more hosts will not happen that frequently. In the event that it does happen, it is true that the device (and, in turn, the user) will have very little information on which to make a decision. In this scenario, the device will most likely choose one host randomly. Then, by comparing the numbers on the intended host and the device, the user can determine whether the device is connected to the correct host.

40. If we consider a digital camera connected to a DWA embedded inside its plastic, the camera by itself is just a USB device. The DWA is the Wireless USB device. But to the person looking at it, they are not going to know the

difference. How do the device rules from the association model spec apply to this scenario?

In this case, the product is being marketed as a Certified Wireless USB device, so it would need to comply with the requirements for devices as outlined in section 3.2 of the Association Model specification.

41. If a Certified Wireless USB device has both a display and a wired USB port also but the manufacturer does not want to implement both the association models, is this acceptable?

If a device has both a USB port and a display, then that device must support both the numeric and cable association models. This requirement ensures that hosts and devices with a Certified Wireless USB logo will always be able to association with each other.

42. In the numeric model, if the numbers displayed by the host and device do not match, how does the user reject the association?

If the product (host or device) features a "cancel association" button, then the user would press that. If the product does not have a cancel feature, then it should eventually timeout. The timeout value is an implementation decision by the manufacturer: It is required to be at least 20 seconds, but has no upper bound.

43. How many connection contexts should a device be able to store?

A device should provide persistent storage for as many different hosts as the user is expected to connect the device to. The Wireless USB 1.0 specification requires devices to support at least one connection context. It is recommended that devices provide storage for more than one connection context however; otherwise, if the device is associated with a second host, the settings for the first host will be overwritten. It is expected that eight connection contexts would be more than enough for most scenarios.

44. If a device runs out of room for new connection contexts, what should it do?

The Wireless USB 1.0 specification requires a device to always accept a new connection context. Devices that run out of space for new connection contexts must make room by removing an existing connection context. There are many different methods of removing connection contexts. One method is to maintain a count of how many times each connection context is used and remove the least frequently used entries first. Another method is to maintain a timestamp of when each connection context was last used and to remove the least recently used entries first.

45. In the numeric model, if the user makes a mistake and clicks "match" on the host and "no match" on the device, then the host will store the connection context in persistent memory. However, the device did not store a copy, and thus the connection context is useless to the host, can never be used, and wastes valuable resources on the host.

It is expected that most hosts will have the resources to store hundreds of connection contexts. However, for hosts with limited resources, this may indeed be an issue. One way to mitigate this issue is to maintain a flag that indicates whether the connection context has ever been used to successfully complete a four-way handshake. If a stored connection context has never been used in a four-way handshake, then it is very likely that something went wrong in the association process and the host can safely remove the connection context from memory. Of course, the host should make sure that a reasonable amount of time has passed in order to give the device a chance to connect with the connection context.

46. I'm trying to verify the test vectors for the numeric model, but I'm having trouble because the numbers are so big!

Arnold Reinhold's Big Number Calculator is a useful tool that was designed to work with large cryptographic calculations. http://world.std.com/~reinhold/BigNumCalc.html

47. The Wireless USB 1.0 specification, section 6.2.4 mandates the use of RSA for PK. The association model supplement only talks of Diffie-Hellman. Does the association model supplement supersede the RSA requirement or are there still scenarios where RSA will need to be supported?

As is indicated in the errata, the RSA encryption type is obsolete.

48. If the association fails for any reason in the Cable Model, should the device stay in the configured state on the USB bus?

This is up to the manufacturer. It is recommended that USB devices connected via the USB cable always act like USB devices, independent of the status of the wireless interface. Thus, in this scenario, the device should continue to operate correctly via the wired port. It would not operate over Certified Wireless USB until the Cable Model association had fully completed. However, any failure of association over the cable model requires that products display a failure notification to the user.

49. On Page 23 of the specification (1.0 release) and also page 22, there is mention of an association timeout. For the timeout in section 5.3.7 on page 23 there is mention of a 20 second minimum, but on page 22 under section 5.3.4 there is no mention of a minimum. Is it the same?

Yes, the timeout referred to informatively in section 5.3.4 is the same timeout defined normatively in section 5.3.7. It may be helpful to think of the timeouts in these terms: After the host and device have each completed section 5.3.4, they both start their independent timeout counters. Each side must wait at least 20 seconds before timing out without user input. Of course, if user input is received before that time, a product is allowed to disregard the timeout minimum.

50. Is there a limit to the maximum number of connection contexts a Wireless USB host could hold in its storage?

There is no theoretical limit to the number of connection contexts that a host can store.

51. Does the host change its CHID on every boot or does it retain it?

It is assumed that a host would never change its CHID value once established. If a host did change its CHID value, then devices that wanted to connect to that host would need to be re-associated.

52. Is it possible to distribute Connection Contexts in some way other than the official methods described in the association models supplement (for example, at manufacture time)?

In the case where a manufacturer wishes to pre-associate a host and a device, it is permissible to configure the hosts and devices with the necessary Connection Context information at manufacture time so that they will work together out-of-the-box for the end user.

However, in order to obtain logo certification, it would still be necessary for a product to implement and support all mandatory association models, in addition to any proprietary ones.

53. The purpose of the Association Models is to establish a connection key (CK) on both the device and host which will secure the 4-way handshake.

Looking at the WiMedia/Wireless USB frame structure, the "session" keys to be used for encryption/decryption are identified by the TKID in the frame. The CK is neither a session key nor seems to have a mechanism to assign a TKID.

Please refer to section 6.5.1 of the Certified Wireless USB specification for an explanation of how the TKID relates to the CK. The CK is a host-supplied value that is established in the device during association. Later, during device connection, the host also supplies the host DevAddr, device DevAddr, the TKID, and a host nonce. The device provides a device nonce. From all of these parameters, the host and device use the key derivation function to calculate the session keys.

54. In section 5.3.1 of the Association Models supplement, it says that devices must search for hosts for at least 30 seconds but not more than 2 minutes. What does this mean exactly? Must a device wait the full 30 seconds to begin association, even if it has discovered a host sooner?

This timeout period only refers to the amount of time a device must search for a host. Once a device has found a host, it may immediately proceed with the rest of the association process; the device does not need to complete the minimum 30 second waiting period.

55. Are there any plans for adding additional association models? What is the process for doing that?

There are no announced plans or roadmaps yet for adding additional association models. However, we realize that as technologies emerge and evolve, that it will become necessary at some point in the future to modify existing association models and add new ones. At that time, the Wireless USB Promoter Group will re-establish the Association Model working group, who will in turn develop the new specification following the same format as was done for the 1.0 release.

56. When a host completes the numeric association procedure with a device, should either the host or device disconnect the link and connect again with the new connect bit set to 0 before moving on to do the 4-way handshake?

There is no need to disconnect after the numeric association. The host must directly proceed to the 4-way handshake after doing the association.

57. How can I test if my random number generation is really generating good random numbers?

RFC 4086 is some good background reading. Also, the NIST project on Random Number Generation and Testing has a good list of tests.

58. How big is a software implementation of Diffie-Hellman?

An ARM implementation optimized for speed is approximately 16kb code size. Optimizations to reduce code size are possible, but at the expense of speed. Philip Zimmermann has published some numbers on the performance and code size of a software implementation of Diffie-Hellman at the following web site: http://philzimmermann.com/EN/bnlib

59. Is there any help available for optimizing the Diffie-Hellman calculations?

The big number arithmetic (in particular, the modular exponentiation operations) used by Diffie-Hellman can be a challenge for software developers unaccustomed to cryptography work.

Philip Zimmermann has a math library available (either under GNU license or non-GNU license) to assist manufacturers. Please see his web site for more information: http://philzimmermann.com/EN/bnlib

60. For the cable model, only the interface descriptor is shown. What is the intention of this? Does the specification encourage a particular device implementation (i.e., composite versus compound device)?

There are three basic ways to build a device that supports the Cable-Based Association Framework (CBAF):

- A composite device where one of the interfaces is CBAF.
- A compound device, where one of the sub-devices supports CBAF (as either a composite device or a single interface device).
- A single interface device whose interface is CBAF.

Do not implement a DWA as a compound device that also works as a hub when connected with a wire because Microsoft's driver does not support this.

61. Some of the association rules for hosts are too restrictive for kiosk scenarios. For example, consider the case of a printer kiosk that wants to allow cameras to connect. Can the rules be relaxed to improve usability?

Kiosks may indeed be a special class of host that may not care if an attacker can associate. For example, considering the example scenario above, the following steps would probably increase the ease of use at the cost of possibly allowing a man-in-the-middle attacker to associate with the host:

1. Kiosk (host) always in "allow new associations" mode (technically a violation of the association rules)
2. Customer walks up with digital camera (device). Device is fully compliant with the association rules.
3. Customer tells camera to look for a new host
4. Camera finds kiosk
5. Camera and kiosk automatically start numeric association method
6. Camera and kiosk both display numeric verification number
7. Customer clicks "ok" on the camera if numbers match
8. Kiosk automatically accepts, assumes that numbers match.
9. Customer proceeds with using the kiosk.

In this case, the kiosk would not be able to be certified since it does not meet the association rules as specified in the Certified Wireless USB specification and test suite. However, certified devices would work fine with the kiosk.

62. The specification appears to tie numeric association to the presence of the "New Connection" bit being set in the DN_CONNECT message from the device. Is this correct? If the host receives a DN_CONNECT("New Connection"=TRUE) message, should it assume Numeric Association is required?

Yes, if a device asserts the new connection bit in a DN_CONNECT, then it is, ipso facto, requesting an in-band association. Currently, the numeric association is the only in-band association method.

63. If a proposal for a new in-band association method were proposed, how could the presenter differentiate this new connection request from a Numeric Association request?

The first field in the numeric association M1 message contains a version field (currently set to 0x01). In the event that a future in-band association method is developed, the version field will be incremented accordingly, along with whatever changes are required to accommodate the new association method.

64. Is the purpose of the Wireless USB security descriptors only to distinguish between encrypted (CCM) and non-encrypted? Why would you not consult the security descriptors to determine an association type for available, new in-band associations? This seems to be contemplated, if not implemented in the descriptors.

The security descriptors were developed very early on in the development of Wireless USB, before the association methods were finalized. In our security philosophy, we decided not to trust any information from the device until after an association had been completed.

65. What is the purpose of the RSA_1 encryption type in the security descriptors?

As is indicated in the errata, the RSA encryption type is obsolete.

66. What is the purpose of the WIRED encryption type in the security descriptors? What is meant by "Virtual encryption provided by the wire"? Is this intended to make a distinction between wired connections vs. unencrypted over-the-air connections?

This information is not used in the current release of the specification.

67. In the numeric association model, does a device have to wait until it receives M4 from the host to begin its DHKey calculations?

A device does not have to wait until it receives M4 from the host to begin calculating DHKey. The device can start computing DHKey as soon as it receives M3. However, the device should not calculate the CK value derived from the DHKey until after M4 has been received.

68. During the numeric association model, if the device accepts the verification digits after the host, is there any possibility of a race condition where the host does not know when the device has finished calculating the keys? On embedded platforms, the time to calculate keys could be very long.

If the device receives a SET_SECURITY_DATA(M4) from the host and the device has not yet completed calculating the CK, the device can respond with a NAK per 5.5.2 of the Wireless USB specification. This informs the host that the device is not ready to proceed yet. The host would then be expected to keep trying for some period of time.

As an example, a possible implementation of the numeric association might proceed as follows (this example assumes the first several steps of the numeric model have already been completed):

1) Host will send M2
2) Device will start calculating DHKey
3) Host will request M3 from the device
4) Host user will confirm the number

5) Host will send M4

6) Device NAKs M4 until step 8 is complete

7) Device user will confirm the number

8) Device computes CK (HMAC-SHA-256)

9) Device ACKs M4

10) 4-way handshake

69. Many of the numbers used in the association model supplement are big endian, but the USB world in general is little endian. How do I tell which attributes should be sent as big/little endian?

The following statements will help clarify the endianness of the association model supplement:

1) The AM spec is believed to be consistent with regard to byte order for transmission of numbers (little-endian) and byte sequences (in-order). However, whether an attribute is considered a number or byte sequence is not clearly defined, leading to considerable confusion.

2) Per section 2.6 of the AM Supplement, all numbers listed in the document are presented in big endian notation. (Example: Decimal 10000 is shown as 0x2710 where memory address location increases from left to right.)

3) Per section 2.6, all numbers are required to be transmitted over USB as little endian. (Example: Decimal 10000 transmitted as 0x10 followed by 0x27)

4) All numbers used for cryptographic calculations must be stored in memory as big endian. (This is not explicitly stated, which is an oversight. However, this can be deduced from the test vectors.)

5) Although not explicitly stated as such, any field that is not a UNICODE string and that is less than or equal to 64 bits in length is considered a number. All numbers must be transmitted over USB as little endian.

6) Any field that is a UNICODE string or that is greater than 64 bits in length is considered a byte array. All byte arrays must be stored in memory from left to right, where the left-most byte is stored in the lowest memory address location. All byte arrays must be transmitted over USB from left to right.

7) Examples of attributes that are numbers and are transmitted as little endian: AssociationTypeId, AssociationSubTypeId, Length, AssociationStatus, LangID, BandGroups, Flags, AssociationTypeInfoSize

8) Examples of attributes that are considered byte arrays: DeviceFriendlyName, HostFriendlyName, CHID, CDID, CK, ConnectionContext, SHA-256(M3), PK_H, PK_D.

9) When packing a buffer for transmission, the offsets listed in the spec govern where each field should start. For example, considering the M2 buffer: Version is sent first (1 byte). Then LangId (2 bytes, little endian). Then HostFriendlyNameLen (byte array). Then PKsubH (384 bytes, byte array). Then HostFriendlyName (byte array). On the receiving end, these will be assembled into a buffer as follows: Version first (1 byte). Then LangId (2 bytes, little endian). Then HostFriendlyNameLen. Then PKsubH (384 bytes, byte array). Then HostFriendlyName (byte array).

10) We will produce a dump of a complete exchange which will make it more difficult to misinterpret.